

Remarks

This REQUEST FOR CONTINUED EXAMINATION and RESPONSE is in reply to the Office Action mailed December 2, 2005. A Petition for Extension of Time is submitted herewith, together with the appropriate fee. No fee is due for the addition of new claims.

I. Summary of Examiner's Rejections

Prior to the Office Action mailed December 2, 2005, Claims 1-26 were pending in the Application. In the Office Action, Claims 1-9, 12-20 and 23-26 were rejected under 35 U.S.C. 103(a) as being unpatentable over Kampe et al. (U.S. Patent No. 6,854,069, hereinafter Kampe). Claims 10, 11, 21 and 22 were rejected under 35 U.S.C. 103(a) as being unpatentable over Kampe in view of Wachtel (U.S. Patent No. 6,847,974), and further in view of Burdeau (U.S. Patent No. 6,868,442).

II. Summary of Applicant's Amendment

The present Response amends Claims 1 and 12; cancel Claims 23-26; and add new Claims 27-30, leaving for the Examiner's present consideration Claims 1-22 and 27-30. Reconsideration of the Application, as amended, is respectfully requested. Applicant respectfully reserves the right to prosecute any originally presented or canceled claims in a continuing or future application.

III. Claim Rejections under 35 U.S.C. §103(a)

In the Office Action mailed December 2, 2005, Claims 1-9, 12-20 and 23-26 were rejected under 35 U.S.C. 103(a) as being unpatentable over Kampe (U.S. Patent No. 6,854,069). Claims 10, 11, 21 and 22 were rejected under 35 U.S.C. 103(a) as being unpatentable over Kampe in view of Wachtel (U.S. Patent No. 6,847,974), and further in view of Burdeau (U.S. Patent No. 6,868,442).

Claim 1

Claim 1 has been amended by the current Response to more clearly define the embodiment therein. As amended, Claim 1 defines

1. *(Currently Amended) A system for high availability clustering of a group of computer nodes, comprising:*

a computer that allows an application to access a set of resources of various resource types, including different application servers, within a cluster, wherein said resources and application servers are available at said computer or at another computer in the cluster, and wherein resources and application servers can be grouped by resource type within a pool of resources;

a cluster server that operates at said computer and that allows access to said set of resources;

a resource interface provided by said cluster server that provides an abstraction layer and allows the cluster server to receive uniform requests from the application and communicate the requests to said set of resources;

a plurality of plugins that are plugged into the resource interface to provide a set of application-specific callbacks from the cluster server to the different resources, wherein the system includes a plugin for each resource type, and wherein each plugin implements a resource API to encapsulate its particular resource type-specific behavior and to isolate the cluster server from that behavior while providing access to its pool of resources;

wherein additional plugins may be plugged into the resource interface for other resource types; and

wherein the system can be extended by adding additional computers with cluster servers and resource interfaces operating thereon.

Claim 1, as currently amended, defines that the resources (e.g. application servers) can be grouped by resource type within a pool of resources. A cluster server allows access to the set of resources using a resource interface, which in turn provides an abstraction layer and allows the cluster server to receive uniform requests from the application and communicate the requests to the resources. The system includes a plugin for each resource type, wherein each plugin implements a resource API to encapsulate its particular resource type-specific behavior and to isolate the cluster server from that behavior, while at the same time providing access to its pool of resources. Applicant respectfully submits that these features are not disclosed or suggested by the cited references.

The advantages of the embodiment defined by Claim 1 include that it provides a uniform high availability framework for use in a cluster that includes different resource types. One of the

problems with currently available clustering systems is the need to integrate the cluster framework with its applications by providing a set of application-specific callbacks. These application-specific callbacks are needed to allow adequate control and monitoring of the software applications running on the cluster. Examples of callbacks include application scripts, DLL's, and regular compiled/executable code. In a traditional cluster, each additional type of application server would require their own specific callback. However, using the present embodiment, each resource instance (e.g. each application server instance) can be of a particular resource type (e.g. the "WLS application server" or "Tuxedo application server" types). Plugins implementing the resource API encapsulate their resource type-specific behavior, and isolate the cluster server from that behavior. The plugins then provide the mapping between framework's resource management abstractions and the resource type-specific way of realizing the particular functionality, without requiring their own specific callback on the cluster server.

Kampe discloses a system and method for achieving high availability (HA) in a networked computer system. As disclosed therein, the networked computer system includes nodes that are connected by a network. The method includes using high-availability-aware components to represent hardware and software in the networked computer system, managing the components to achieve a desired level or levels of redundancy, and monitoring the health of the networked computer system, including the health of the components and the nodes. (Column 2, lines 38-43). An HA system may be comprised of many independent "managed components." Such components may include: (1) hardware "field replaceable units" ("FRUs"), which can be powered on and off, removed, or replaced without affecting the operation of other hardware components and (2) independent software functions ("logical FRUs"), which can be shut-down, restarted, or upgraded without affecting other software in the system. Components may be "CPU nodes," each of which is running an instance of an operating system. Using "CPU nodes," the system enables individual nodes to be rebooted with little or no effect on other nodes. Further, hardware components may be distributed across multiple independent shelves and buses so the failure of a single fan, power-supply, CPU, or even a backplane does not take out the entire cluster or networked computer system. (Column 6, lines 35-62). A hot-swap manager ("HSM") may provide a higher-level service

that runs above hot-swap bus managers, such as cPCIs, and managers for other types of FRU interconnects, such as an application that watches for payload cards to come alive on dedicated serial links or join an ethernet. The underlying bus managers are typically responsible for detecting and identifying newly inserted FRUs. The HSM may be responsible for deciding what to do with the new device. The HSM preferably knows about types of supported FRUs through code and/or configuration data, for example. For each supported FRU type, the HSM preferably knows which component managers are to be instantiated to make the FRU useful. (Column 16, lines 44-61).

In the Office Action it was submitted that Kampe discloses that components may be application servers, and that a plurality of plugins can be plugged into system to provide a mapping between the systems resource management functions and resource type-specific (i.e. application server-specific functionality). However, the above description appears to suggest that, in Kampe, while each node may be running an instance of a particular *operating system*, there doesn't seem to be any discussion of, for example, multiple types of operating systems, or of multiple types of different application servers running on those operating systems. By contrast, as defined by Claim 1, the system therein allows a client application to access a set of resources of various resource types, including *different application servers*, (e.g. WebLogic servers and Tuxedo servers). The system includes a plugin for each *type* of application server, and the plugin implements a resource API to encapsulate its particular application-server-specific behavior from the cluster server, so that the cluster server can instead provide a uniform interface to the client application.

Furthermore, as disclosed by Kampe, a HSM is responsible for deciding what to do with a new device, in that the HSM preferably knows about types of supported FRUs through code and/or configuration data, and preferably knows which component managers are to be instantiated to make the FRU useful. The above description appears to suggest that, in Kampe, the system must incorporate specific code into the server to support each FRU type, which appears analogous to incorporating application-specific callbacks in the server itself. However, as defined by Claim 1, a plurality of plugins are plugged into a resource interface, which in turn provide a set of application-specific callbacks, but otherwise isolates the cluster server from the application-specific behavior of the different application servers and resources.

- 11 -

Attorney Docket No.: BEAS-1119US1
kfk/beas/1119us1/1119us1.RCE.12.02.05.wpd

REST AVAILABLE COPY

In view of the above comments, Applicant respectfully submits that Claim 1, as amended, is neither anticipated by, nor obvious in view of the cited references, and reconsideration thereof is respectfully requested.

Claim 12

Claim 12 has been amended similarly to Claim 1 to more clearly define the embodiment therein. Applicant respectfully submits that Claim 12, as amended, is likewise neither anticipated by, nor obvious in view of the cited references, and reconsideration thereof is respectfully requested.

Claims 2-11 and 13-26

Claims 23-26 have been canceled, rendering moot the rejection of these claims. Claims 2-11 and 13-22 are not addressed separately, but it is respectfully submitted that these claims are allowable as depending from an allowable independent claim, and further in view of the comments provided above. Applicant respectfully submits that Claims 2-11 and 13-22 are similarly neither anticipated by, nor obvious in view of the cited references, and reconsideration thereof is respectfully requested.

IV. Additional Amendments

Claims 27-30 have been newly added by the present Response. Applicant respectfully requests that new Claims 27-30 be included in the Application and considered therewith.

V. Conclusion

In view of the above amendments and remarks, it is respectfully submitted that all of the claims now pending in the subject patent application should be allowable, and reconsideration thereof is respectfully requested. The Examiner is respectfully requested to telephone the undersigned if he can assist in any way in expediting issuance of a patent.

Enclosed is a PETITION FOR EXTENSION OF TIME UNDER 37 C.F.R. §1.136 for extending the time to respond up to and including June 2, 2006.

- 12 -


Attorney Docket No.: BEAS-1119US1
kfk/beas/1119us1/1119us1.RCE.12.02.05.wpd

Application No.: 10/693,137
Response to OA dated: December 2, 2005
Response dated: June 2, 2006

The Commissioner is authorized to charge any underpayment or credit any overpayment to Deposit Account No. 06-1325 for any matter in connection with this response, including any fee for extension of time, which may be required.

Respectfully submitted,

Date: June 2, 2006

By: 

Karl Kenna
Reg. No. 45,445

Customer No.: 23910
FLIESLER MEYER LLP
Four Embarcadero Center, Fourth Floor
San Francisco, California 94111-4156
Telephone: (415) 362-3800

- 13 -

Attorney Docket No.: BEAS-1119US1
kfk/beas/1119us1/1119us1.RCE.12.02.05.wpd